Analisis Perbandingan *Quality of Service* (QoS) Penerapan Snort IDS dan Bro IDS Dalam Arsitektur *Software Define Network* (SDN)

Hendrawan[1], Parman Sukarno[2], Muhammad Arief Nugroho[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1,2,3]Kelompok Keahlian Telematika
[1]wanhendra@students.telkomuniversity.ac.id, [2]psukarno@telkomuniversity.ac.id,
[3]arief.nugroho@telkomuniversity.ac.id

Abstrak

***Intrusion Detection System* (IDS) merupakan sebuah aplikasi yang ditujukan menjadi pemantau aktivitas jaringan atau sistem dan dapat menemukan jika terjadi operasi yang berbahaya. Penerapan IDS pada arsitektur *Software Define Network* (SDN) memiliki salah satu kekurangan. IDS dalam arsitektur SDN dapat menimbulkan penurunan *Quality of Service* (QoS). Sehingga jaringan tidak dapat memberikan layanan terhadap lalu lintas jaringan. *Throughput, delay,* dan *packet loss* merupakan parameter penting dalam pengukuran QoS. Snort IDS dan bro IDS merupakan *tools* penerapan IDS pada jaringan. Keduanya memiliki perbedaan, salah satunya terdapat pada metode deteksinya. Snort IDS menggunakan metode *signature based detection* sedangkan bro IDS menggunakan metode *anomaly based detection*. Perbedaan keduanya secara tidak langsung memiliki pengaruh dalam menangani lalu lintas jaringan yang melewatinya. Dalam pemilihan *tools* IDS untuk deteksi serangan tentunya akan dipilih *tools* yang terbaik untuk diterapkan. Dalam penelitian ini dibandingkan kedua *tools* tersebut. Perbandingan dilakukan dengan pengujian parameter *throughput, delay, packet loss, CPU usage,* dan *memory usage*. Dari pengujian didapatkan bahwa bro IDS lebih baik dibandingkan snort IDS dengan hasil pengujian 4:1 untuk parameter *throughput*, 3:1 untuk parameter *delay*, dan 5:1 untuk parameter *packet loss*. Akan tetapi, pada parameter CPU *usage* dan *memory usage* Bro IDS membutuhkan *resource* yang lebih besar dibandingkan snort IDS.**

Kata kunci : SDN, QoS, IDS, Snort, Bro.

Abstract

Intrusion Detection system (IDS) was an application which was aimed to monitor network activity or system and it could find if there was a dangerous operation. Implementation of IDS on Software Define Network architecture (SDN) had one deficiency. IDS on SDN architecture might generate a decrease of Quality of Service (QoS). So the network could not provide services to the existing network traffic. Throughput, delay and packet loss were important parameters of QoS measurement. Snort IDS and bro IDS were tools in the application of IDS on the network. Both had differences, one of which was found in the detection method. Snort IDS used a signature based detection method while bro IDS used an anomaly based detection method. The difference between them had effects in handling the network traffic through it. In this research, we compared both tools. This comparison are done with testing parameters such as throughput, delay, packet loss, CPU usage, and memory usage. From this test, it was found that bro IDS was better than snort IDS with 4:1 test result for throughput parameters, 3:1 for delay parameters, and 5:1 for packet loss parameters. However, CPU usage and memory usage parameters requires more resources than snort IDS.

Keywords: SDN, QoS, IDS, Snort, Bro.

1. Pendahuluan

Latar Belakang

Sofware Define Network (SDN) merupakan suatu paradigma teknologi jaringan terbaru yang menggunakan pengendalian terpusat (*Controller*) [18]. Akan tetapi SDN tidak dapat menghindari serangan atau ancaman pada jaringan. Dalam perkembangannya beberapa serangan dapat terjadi dengan *trend* penyerangan yang berubah-ubah [4][5][6]. Dengan isu penyerangan tersebut dibutuhkan *Intrusion Detection System* (IDS). IDS adalah sebuah aplikasi yang ditujukan menjadi pemantau aktivitas jaringan atau sistem dan dapat menemukan jika terjadi operasi yang berbahaya [17].

Penerapan IDS pada jaringan SDN memiliki beberapa kelebihan, yaitu akan membuat jaringan SDN menjadi lebih aman, SDN sangat efektif untuk *monitoring traffic* karena SDN memiliki kontrol langsung atas seluruh jaringan, ancaman penyerangan dapat ditemukan segera, dan penerapan IDS akan lebih mudah untuk satu atau banyak *interface device* (*fleksibel*) [12]. Akan tetapi, penerapan IDS pada jaringan SDN memiliki beberapa kekurangan, yaitu IDS dapat menambahkan alur pemrosesan *trafic* jaringan dikarenakan IDS merupakan *application layer* pada jaringan SDN [19] dan akan terjadi penurunan *Quality of Service* (QoS) yang dibuktikan pada penelitian [10] dengan menggunakan snort IDS sebagai *tools* untuk IDS. Snort IDS merupakan *tools Network Intrusion Detection System* (NIDS) yang paling populer dan sangat mudah digunakan untuk berbagai jenis jaringan [2][8][16], snort memiliki tiga kegunaan utama yaitu sebagai *sniffer,* IDS, dan *Intrusion Prevention System* (IPS) [16].

Snort IDS menggunakan metode *signature based detection* [2]. Metode *signature based detection* (*Packet Based*) bekerja dengan mendefinisikan perilaku atau serangan dengan seperangkat aturan atau pola penyerangan didalam *database* [12]. Kelebihan *signature based detection* adalah proses yang handal dan detail dalam proses deteksi [12] dan mudah diimplementasikan dalam penulisan *rule*. Namun *signature based detection* memiliki beberapa kekurangan yaitu tidak dapat mendeteksi serangan yang baru dikarenakan signature tidak terdapat di *database rule*, memerlukan waktu dalam proses deteksi karena setiap *packet* harus dibandingkan oleh semua *signature* [12], dan *monitoring packet* pada jaringan yang besar sangat tidak dianjurkan [2].

Selain *signature based detection* terdapat salah satu metode deteksi lainnya dalam IDS yaitu *anomaly based detection* [2]. *Anomaly based detection* bekerja berbasis statistik, perilaku, dan data. Statistik melibatkan perilaku pengguna dalam periode waktu tertentu. Kelebihan *anomaly based detection* adalah dapat mendeteksi tipe serangan baru dan dapat *monitoring* pada jaringan yang besar [12][2]. Sedangkan *anomaly based datection* memiliki beberapa kekurangan yaitu memerlukan *resource* yang lebih besar dan memungkinkan terjadi kesalahan dalam menghasilkan alarm [12]. Bro IDS merupakan salah satu *tools* NIDS yang menggunakan *anomaly based detection*. Bro IDS memberikan fasilitas kepada pengguna untuk melakukan percobaan penelitian yang mengembangkan cara pandang baru terhadap data (*anomaly*).

Topik dan Batasannya

Topik yang diangkat dalam penelitian ini adalah pengaplikasian snort IDS dengan metode *signature based detection* dan bro IDS dengan metode *anomaly based detection* pada jaringan SDN dapat mempengaruhi QoS (*throughput*, *delay*, dan *packet loss*), CPU *usage* dan *memory usage*. Adapun pada penelitian ini menggunakan ryu sebagai SDN *controller* yang diaplikasikan didalam *emulator* mininet untuk simulasi jaringan SDN dan menggunakan *traffic generator* (iperf). Penelitian ini juga tidak berfokus pada *false positive* dan *false negative* dalam kinerja IDS. Penerapan rules berjumlah 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 yang digunakan untuk deteksi serangan pada layer tiga, empat, dan tujuh pada osi layer. Pada pengujian juga dilakukan *generate traffic* sebesar 0, 10, 20, 30, 40, dan 50 Mbps pada jaringan yang dibuat.

Tujuan

Tujuan dari penelitian ini dapat membandingkan kinerja parameter *throughput*, *delay*, *packet loss*, CPU *usage*, dan *memory usage* saat pengaplikasian snort IDS dan bro IDS yang berperan sebagai pemantau aktivitas dari arsitektur SDN. Hal ini bisa diilustrasikan pada Tabel 1.

Tabel 1. **Keterkaitan antara tujuan, pengujian dan kesimpulan**

No	Tujuan	Pengujian	Kesimpulan
1	Melihat pengaruh jumlah *rules* pada kinerja parameter *throughput*, *delay*, dan *packet loss* saat pengaplikasian snort IDS dan bro IDS pada arsitektur SDN.	Mengaplikasikan snort IDS dan bro IDS dengan jumlah *rules* 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 pada arsitektur SDN serta melihat kinerja parameter *throughput*, *delay*, dan *packet lost* dengan cara *generate traffic* (iperf).	Evaluasi kinerja *throughput*, *delay*, dan *packet loss* terhadap jumlah *rules* pada snort IDS dan bro IDS dalam arsitektur SDN.
2	Melihat pengaruh *background traffic* pada kinerja parameter *throughput*, *delay*, dan *packet loss* saat pengaplikasian snort IDS dan bro IDS pada arsitektur SDN.	Mengaplikasikan snort IDS dan bro IDS dengan *background traffic* 0, 10, 20, 30, 40, 50 Mbps pada arsitektur SDN serta melihat kinerja parameter *throughput*, *delay*, dan *packet loss* dengan cara generate traffic (iperf).	Evaluasi kinerja *throughput*, *delay*, dan *packet loss* pada snort IDS dan bro IDS dalam arsitektur SDN yang memiliki *background traffic* 0, 10, 20, 30, 40, 50 Mbps.
3	Melihat pengaruh penggunaan CPU saat pengaplikasian snort IDS dan bro IDS pada arsitektur SDN.	Mengaplikasikan snort IDS dan bro IDS pada arsitektur SDN serta melihat kinerja CPU.	Evaluasi kinerja CPU pada penerapan snort IDS dan bro IDS dalam arsitektur SDN.
4	Melihat pengaruh penggunaan *memory* saat pengaplikasian snort IDS dan bro IDS pada arsitektur SDN.	Mengaplikasikan snort IDS dan bro IDS pada arsitektur SDN serta melihat kinerja *memory*.	Evaluasi kinerja *memory* pada penerapan snort IDS dan bro IDS dalam arsitektur SDN.

Organisasi Tulisan

Pada jurnal tugas akhir terdapat studi terkait yang mendukung penulisan atau pengerjaan tugas akhir, menjelaskan sistem yang dibangun, evaluasi yang menjawab tujuan dari tugas akhir ini, dan dapat menarik kesimpulan.

2. Studi Terkait

Pada penelitian [10], dilakukan penelitian bahwa *Intrusion Detection System* (IDS) dapat diintegerasikan dalam arsitektur *Software Define Network* (SDN). Pengunaan snort sebagai tools IDS pada penelitian tersebut. Snort memiliki metode *signature based detection* didalamnya. Inti dari penelitian tersebut adalah melihat performansi dari pengaplikasian snort IDS pada jaringan SDN. Pengukuran terhadap parameter *delay*, *throughput*, *jitter*, dan *packet loss ratio* mengalami penurun performansi saat arsitektur SDN mengaplikasikan snort IDS. Saat sebelum pengaplikasian snort IDS pada arsitektur SDN parameter *delay* = 0,084790 s, *throughput* = 4868,721788 Kbps, *jitter* = 0,000594 s, dan *packet loss* = 0%. Sedangkan sesudah pengaplikasian sesudah pengaplikasian snort IDS pada arsitektur SDN parameter *delay* = 0,105761, *throughput* = 4828,444506 Kbps, *jitter* = 0,000774 s, dan *packet loss* = 0%.

Pada penelitian [14], dilakukan penelitian dengan memanfaatkan logika fuzzy dalam proses pemblokiran dengan *adaptive* IPS (snort). Dengan pengaplikasian snort untuk proses deteksi dan pemblokiran menyebabkan nilai *throughput* turun 9,698 Gbps dan naiknya *delay* sebesar 1007,809 ms pada sistem yang dibuat.

Pada paper [2], keamanan jaringan merupakan hal yang sangat berpengaruh signifikan dalam pengelolaan jaringan. IDS merupakan solusi untuk keamanan jaringan. Banyaknya *tools* dalam pengaplikasian IDS. Setiap tools memiliki perbedaan untuk studi kasus yang berbeda-beda. Maka dari itu dilakukan perbandingan antara *Network Intrusion Detection System* (NIDS) *tools*. *Tools* snort, suricata, dan bro dilakukan perbandingan antara ketiganya. Snort, suricata, dan bro memiliki fungsi yang berbeda dan keduanya memiliki perilaku berbeda dalam arsitekturnya. Bro memiliki cara pandang baru dalam hal deteksi yang dikenal dengan *anomaly based*. Sedangkan snort dan suricata menggunakan *signature based*. Tabel 2 adalah gambaran lengkap untuk perbandingan antara snort, suricata, dan bro.

Pada paper [13], *throughput* jaringan tumbuh pada tingkat ekponensial. produk IDS/IPS dapat meningkatkan kekuatan pemrosesan mereka melalui ASICs dan spesifikasi *hardware*. Inovasi dan pendekatan yang berbeda-beda dikembangkan oleh masing-masing *tolls* IDS/IPS. Tantangan pada kecepatan jaringan yang bervolume tinggi merupakan masalah penting. Snort masih konsisten dengan *single-threaded product*. Suricata dengan memanfaatkan peraturan yang dimiliki snort dengan penambahan *multi threading*. Bro menawarkan fitur tambahan melalui *script-based* untuk sistem analisis dan kemampuan menambahkan respon dengan *script* dengan single threading product.

Tabel 2. *Comparative Study of NIDS Tool* [2]

Parameters	Snort	Suricata	Bro
Supported Platform	Win, MacOS, Unix	Win, MacOS, Unix	Unix like system, MacOS
License	GNU GPL V2	GNU GPL V2	BSD
IPS feature	Yes	Yes	No
PGP signed	Yes	Not Applicable	No
Support to high speed network	Medium	High	High
Configuration GUI	Yes	Yes	No
Offline Analysis	Yes for multiple files	Yes for single files	Yes for single files
Threads	Single Thread	Multithreaded	Single Thread
IPV6	Yes	Yes	No
Installation and Deployment	Easy	Easy	Difficult

2.1 Software Define Network

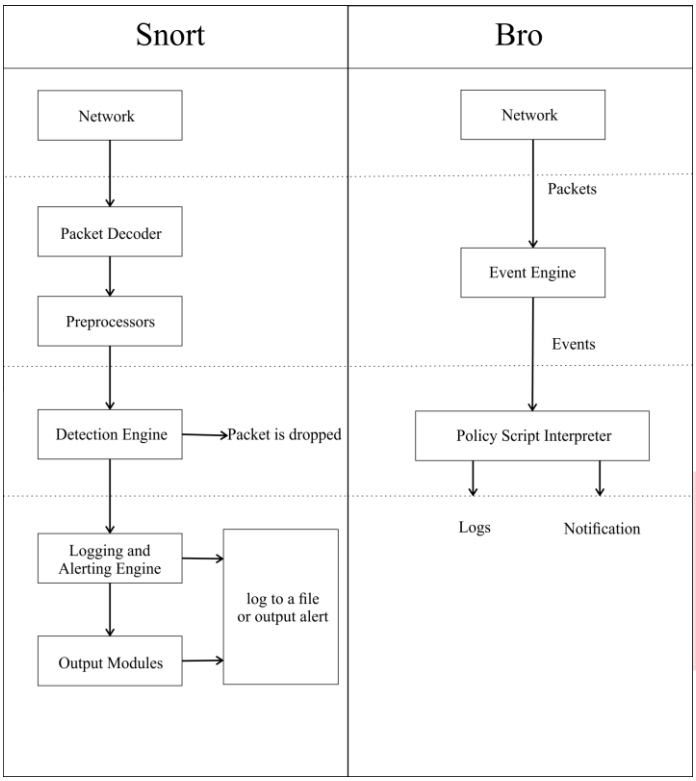
Software Define Network adalah konsep dasar yang melakukan pemisahan antara *control layer*, *infrastructure layer atau forwarding plane*, dan *application layer*. Untuk menghubungkan ketiga layer, SDN menggunakan protokol *openflow* [15]. Openflow dapat bekerja di dua *interface* yaitu *southbound interface* dan *northbound interface*. *Southbound interface* menghubungkan antara *control layer* dan *infrastructure layer*, sedangkan *nortbound interface* menghubungkan *control layer* dan *application layer* [11].

2.2 *Intrusion Detection System* (IDS)

IDS adalah sebuah aplikasi yang ditujukan menjadi pemantau aktivitas jaringan atau sistem dan dapat menemukan jika terjadi operasi yang berbahaya [17]. Pada dasarnya, IDS ada dua tipe dalam pengumpulan informasi saat pemantauan jaringan yaitu *Host-Based* IDS (HIDS) dan *Network-Based* IDS. HIDS memindai dan menganalisa sistem file modifikasi, *log* dari aplikasi, *system calls* dan aktivitas *host* lainnya. Sedangkan NIDS memindai lalu lintas jaringan ke proses mengidentifikasi aktivitas serangan yang bertujuan masuk ke komputer, penolakan layanan serangan, pemindaian port dengan menggunakan pendekatan seperti packet sniffing untuk mengumpulkan data lalu lintas jaringan [9].

Setelah mengumpulkan data dari kegiatan dan kejadian di jaringan, selanjutnya akan diproses untuk mendeteksi sebuah serangan atau tidaknya. Hal ini dilakukan dengan analisis IDS [2] atau metodologi deteksi. Terdapat dua metodologi deteksi IDS yaitu *signature based detection* dan *anomaly based detection*. *Signature based detection* (*Packet Based*) bekerja dengan mendefinisikan perilaku atau serangan dengan seperangkat aturan atau pola penyerangan didalam *database*. Sedangkan *Anomaly based detection* (*Flow Based*) bekerja berbasis statistik, perilaku, atau data. Statistik melibatkan perilaku pengguna dalam periode waktu tertentu [12].

Terdapat beberapa *tools* untuk menerapkan NIDS yaitu Snort IDS dan Bro IDS. Snort IDS mampu untuk mendeteksi *live traffic* pada jaringan, memasukkan *packet*, dan sebagainya. Snort IDS mendeteksi dengan metode *signature based detection*. Snort IDS dapat dijalankan di beberapa sistem operasi seperti Linux, Mac OS X, FreeBSD, Open BSD, UNIX, dan Windows. Snort IDS bisa dioperasikan dalam *mode* IDS dan IPS juga [1]. Pemrosesan snort IDS menggunakan *single threaded engine* dengan infrastruktur snort IDS yang dijelaskan pada Gambar 1 bagian snort IDS. Berikut adalah pemrosesan dalam snort IDS:



Gambar 1. Infrastruktur Snort dan Bro [2]

1. *Packet Capture Library*

   Menangkap *packet* dari satu atau beberapa *interface* jaringan. Untuk sistem operasi linux dan UNIX menggunakan libpcap *library* dan untuk sistem Windows menggunakan WinPcap.

2. *Packet Decoder*

   Pada tahap ini memeriksa *header packet* dan memeriksa kekhasan. *Packet* data kemudian didekripsi untuk diproses lebih lanjut.

3. *Preprocessors*

   Mengumpulkan TCP *stream* dan men*decrypts* HTTP URI dan melanjutkan pada proses *detection engine*.

4. *Detection Engine*

   Proses pengecekan paket-paket terhadap *rule* yang telah dibuat.

5. *Output plugins*

   Menghasilkan keluaran final dengan analisis *logs* dan peringatannya.

Bro IDS juga mampu mendeteksi dan analisis lalu lintas jaringan. Bro IDS mendeteksi dengan metode *anomaly based detection*. Bro IDS sepenuhnya sebagai IDS saja, berbeda dengan snort IDS yang bisa berfungsi sebagai *Intrusion Prevention System*. Bro IDS dapat dijalankan di sistem operasi UNIX [3]. Pemrosesan bro IDS menggunakan *single threaded engine* dengan infrastruktur bro IDS yang dijelaskan pada Gambar 1 bagian bro IDS. Berikut adalah pemrosesan dalam bro IDS:

1. Libpcap

   Bro menggunakan libpcap untuk proses *capture packets* didalam jaringan [3].

2. *Event Engine*

   Menerima lalu lintas yang di *filter* dari libpcap. Proses ini melakukan berbagai pemeriksaan integritas untuk memastikan bahwa paket sudah terbentuk dengan baik, dan memverifikasi *header* IP checksum. Untuk proses ini IP fragmen diletakkan bersama-sama karena *analyzer* mampu mengakses keseluruhan paket IP untuk mengirimkan event ke *policy layer*.

3. *Policy Script Interpreter*

   kebijakan atau (*rules*) ditulis dalam *bro scripting* dengan metode yang tidak bergantung pada deteksi *signature* tradisional. Bro menggunakan metode menganalisis jaringan menggunakan deteksi anomali.

2.3 Quality of Service

Pengukuran *Quality of Service* (QoS) mengacu pada kemampuan jaringan yang dipilih melalui berbagai teknologi dengan beberapa parameter penting diantaranya *throughput*, *delay*, dan *packet loss* [7].

2.3.1 Throughput

Throughput adalah banyaknya paket data yang dapat dikirim per-satuan waktu. Semakin tinggi nilai *throughput*, maka semakin baik jaringan yang dibangun. Faktor yang mempengaruhi *throughput* adalah jenis perangkat, topologi, jumlah *host*, jenis paket data, dll. Rumus untuk mencari *throughput* sebagai berikut:

$$Throughput(bps) = \frac{JumlahDataYangDikirim(bit)}{WaktuPengirimanData(seconds)} \quad (1)$$

2.3.2 Delay

Delay adalah waktu yang dibutuhkan suatu paket data dari awal pengiriman hingga sampai ke tujuan. Semaikin tinggi nilai *delay*, maka semakin buruk jaringan yang dibangun. Faktor yang mempengaruhi *delay* adalah jarak antar *node*, topologi, kepadatan penggunaan jaringan, dan ukuran paket data. Rumus untuk mencari *delay* sebagai berikut:

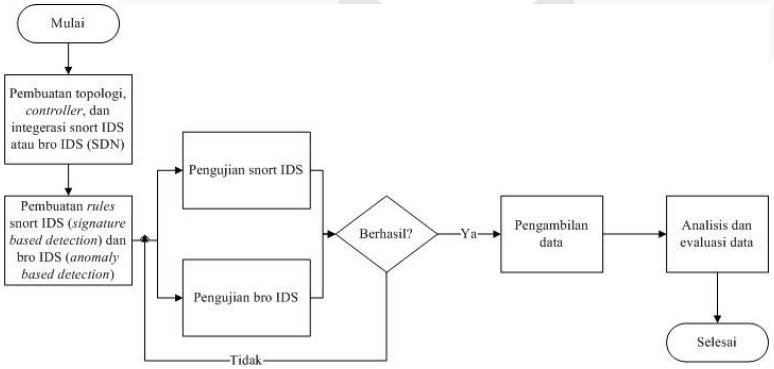
$$Delay(seconds) = WaktuPaketDiterima - WaktuPaketDikirimkan \quad (2)$$

2.3.3 Packet Loss

Packet loss adalah kejadian dimana terdapat kegagalan dalam sebagian paket data ke tujuan di jaringan. Semakin tinggi *packet loss*, maka semakin buruk jaringan yang dibangun. Faktor yang mempengaruhi *packet loss* adalah paket data yang rusak, tingginya *traffic*, dan hilangnya sinyal jika menggunakan jaringan bukan wired. Rumus untuk mencari *packet loss* sebagai berikut:

$$PacketLoss(\%) = \frac{\sum PaketDataYangDikirim - PaketDataYangDiterima}{PaketDataYangDikirim} * 100\% \quad (3)$$

3. Sistem yang Dibangun



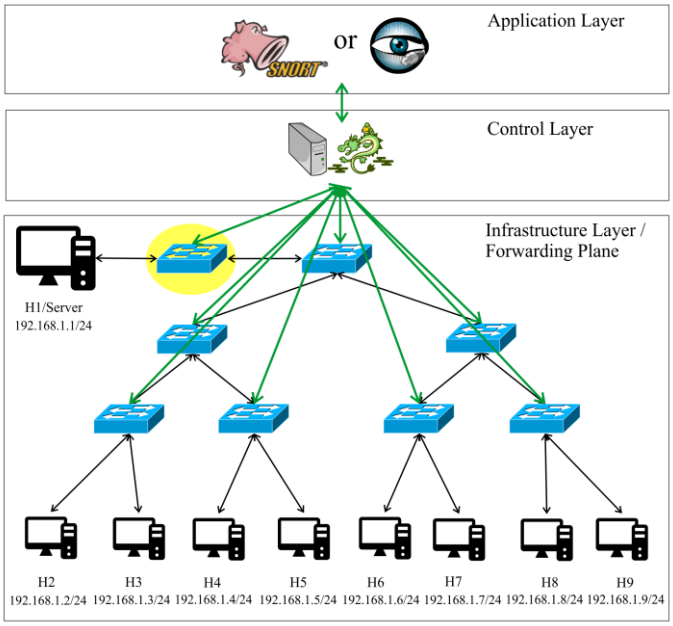
Gambar 2. Tahapan Penelitian

Secara garis besar, sistem yang akan dibangun adalah sistem *Network Intrusion Detection System* (NIDS) untuk mendeteksi *real traffic* yang akan diterapkan pada arsitektur *Software Define Network* (SDN). IDS yang akan diterapkan adalah snort IDS dan bro IDS. Pemilihan kedua *tools* tersebut dikarenakan adanya perbedaan motode deteksinya. *Signature based detection* untuk snort IDS dan *anomaly based detection* untuk bro IDS. Perbedaan tersebut akan dilakukan pengujian terhadap *Quality of Service* (QoS) dengan parameter *throughput*,

delay, dan *packet loss*. Dalam snort IDS dan bro IDS akan terdapat *rules* untuk mendeteksi beberapa serangan atau ancaman pada *layer* tiga, empat dan tujuh di osi *layer*. Pada sistem yang dibangun ini tidak berfokus pada sistem pemblokiran atau *Intrusion Prevention System* untuk langkah berikutnya setelah proses deteksi. Pengujian akan dilakukan melihat pengaruh jumlah *rules* dan *background traffic* pada penerapan IDS dalam arsitektur SDN yang dibuat untuk melihat kinerja snort IDS dan bro IDS. Jumlah *rules* dalam sistem untuk pengujian adalah 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 yang akan disimulasikan pada jaringan dengan *background traffic* 0, 10, 20, 30, 40, dan 50 Mbps dari *bandwidth* yang ditetapkan = 100 Mbps.

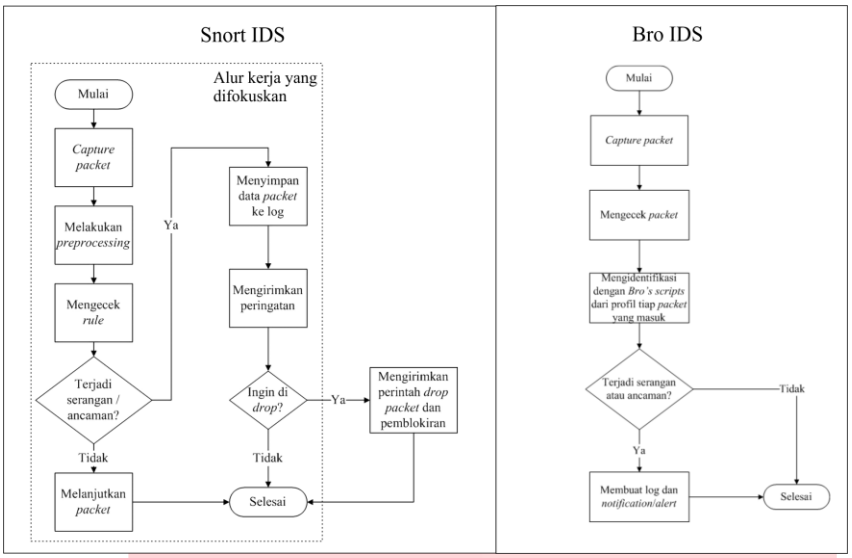
3.1 Topologi yang Dibangun

Pada penelitian ini, mininet digunakan untuk membuat topologi berbasis SDN. Pada Gambar 3 topologi yang digunakan adalah *three tier* dengan pertimbangan topologi yang sering digunakan untuk *data center*. Mininet mendukung pembuatan *virtual switch openflow* yang digunakan dalam arsitektur *Software Define Network* (SDN). Terdapat delapan *switch* pada topologi dengan satu *switch* yang diperuntukkan menghubungkan H1/server dengan *core switch*. Dibangun delapan *client* sebagai *host biasa* atau bisa berfungsi juga sebagai host serangan (*attacker*). Setiap switch akan berkomunikasi dengan *controller* yang dimana pada penelitian ini menggunakan ryu *controller*. Inti dalam penelitian ini yaitu menambahkan *application layer* dengan menerapkan snort IDS dan bro IDS sebagai *tools* untuk deteksi lalu lintas jaringan SDN. Snort IDS atau bro IDS akan diterapkan pada semua interface *switch* yang menghubungkan H1/server dengan *core switch*. Adapun gambaran lengkap topologi dijelaskan pada Gambar 3.



Gambar 3. Topologi

3.2 Alur Kerja Snort IDS dan Bro IDS Pada Sistem



Gambar 4. Alur Kerja Snort IDS dan Bro IDS

Secara garis besar penjelasan alur kerja pada Gambar 4 sama dengan Gambar 1 pada bab sebelumnya. Pada subbab ini menjelaskan snort IDS tidak difokuskan kepada proses *drop packet* setelah proses deteksi *traffic* dan terdapat penjelasan dari perbedaan metode deteksi.

3.3 Skenario Pengujian

3.3.1 Skenario Uji *Quality of Service* (QoS)

Pengujian dilakukan untuk mengetahui pengaruh integerasi snort IDS dan bro IDS pada jaringan SDN dari segi *Quality of Service* (QoS). Parameter yang digunakan adalah *throughput*, *delay*, dan *packet loss*. Pengujian dilakukan dengan cara menerapkan snort IDS dan bro IDS pada jaringan SDN serta melakukan pembangkitan *background traffic*=x menggunakan iperf *tools*. Dengan nilai x={ 10,20,...,50} Mbps. Setiap nilai x pada pembangkitan *background traffic* akan dilihat pengaruh jumlah *rules* atau *scripts detection* pada snort IDS dan bro IDS dengan jumlah *rules* atau *scripts*=y. Dengan nilai y= {0,10,20,...,100}.

3.3.2 Skenario Uji CPU *usage*

Pengujian dilakukan untuk mengetahui penggunaan CPU saat pengaplikasian snort IDS dan bro IDS pada sebuah host. Pada pengujian ini, snort IDS dan bro IDS tidak melakukan *load rules* atau *load scripts* untuk deteksi serangan. Pengujian menggunakan *tools system monitor*.

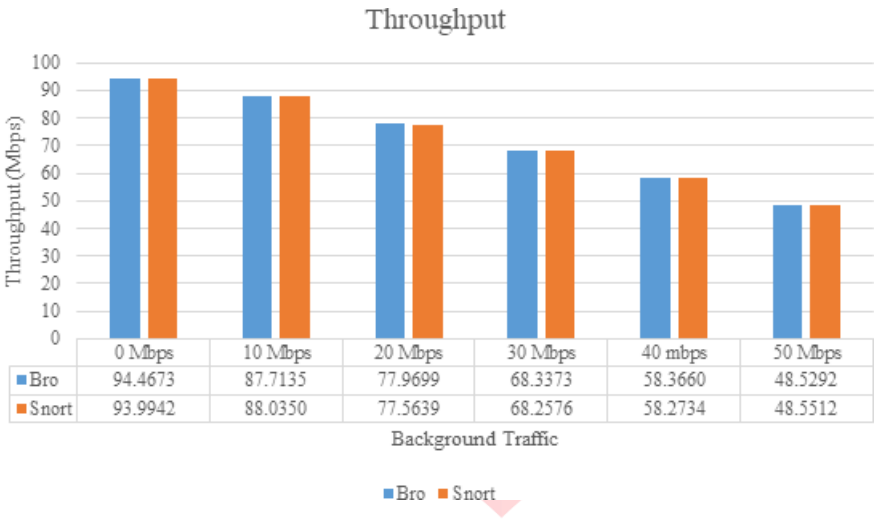
3.3.3 Skenario Uji *Memory usage*

Pengujian dilakukan untuk mengetahui penggunaan memory saat pengaplikasian snort IDS dan bro IDS pada sebuah host. Pada pengujian ini, snort IDS dan bro IDS tidak melakukan *load rules* atau *load scripts* untuk deteksi serangan. Pengujian menggunakan *system monitor tools*.

4. Evaluasi

4.1 Hasil dan Analisis Pengujian

4.1.1 Throughput



Gambar 5. Grafik Hasil Pengujian *Throughput*

Dari hasil penelitian yang telah didapatkan pada Gambar 5 dapat dilihat hasil pengujian sistem terhadap parameter *throughput*. Pada parameter *throughput* semakin besar *throughput* dalam suatu jaringan maka semakin baik kinerja jaringan yang dibuat. Pengujian dilakukan dengan menerapkan snort IDS dan bro IDS dengan jumlah *rules* 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 yang disimulasikan pada jaringan dengan *background traffic* 0, 10, 20, 30, 40, 50 Mbps dari *bandwidth* yang ditetapkan = 100 Mbps.

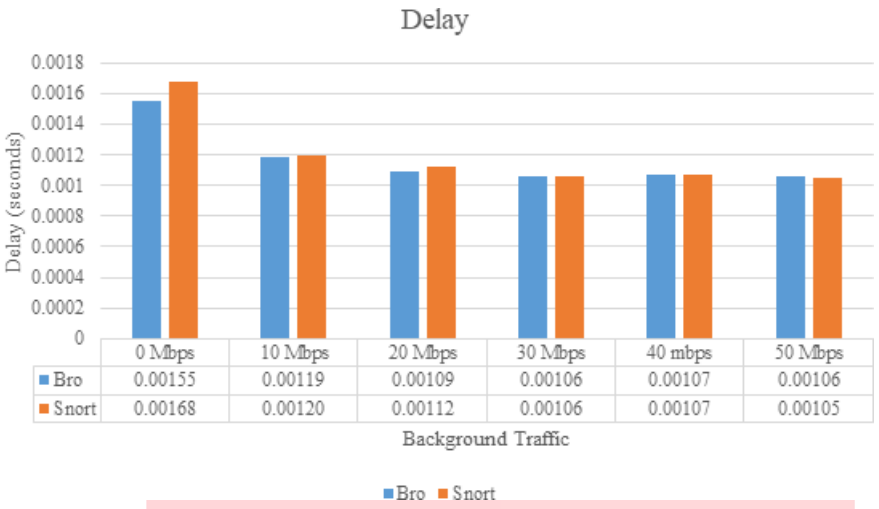
Rata-rata hasil parameter *throughput* dengan *background traffic* 0 Mbps didapatkan hasil 94,4673 Mbps untuk bro IDS dan 93,9942 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,4731 Mbps. Rata-rata hasil parameter *throughput* dengan *background traffic* 10 Mbps didapatkan hasil 87,7135 Mbps untuk bro IDS dan 88,0350 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih buruk daripada snort IDS dengan selisih 0,3035 Mbps.

Rata-rata hasil parameter *throughput* dengan *background traffic* 20 Mbps didapatkan hasil 77,9699 Mbps untuk bro IDS dan 77,5639 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,4060 Mbps. Rata-rata hasil parameter *throughput* dengan *background traffic* 30 Mbps didapatkan hasil 68,3373 Mbps untuk bro IDS dan 68,2576 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0797 Mbps.

Rata-rata hasil parameter *throughput* dengan *background traffic* 40 Mbps didapatkan hasil 58,3660 Mbps untuk bro IDS dan 58,2734 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0926 Mbps. Rata-rata hasil parameter *throughput* dengan *background traffic* 50 Mbps didapatkan hasil 48,5292 Mbps untuk bro IDS dan 48,5512 Mbps untuk snort IDS, dari hasil tersebut *throughput* pada bro IDS lebih buruk daripada snort IDS dengan selisih 0,0220 Mbps.

Dari enam percobaan yang dilakukan terdapat empat percobaan dengan *background traffic* 0, 20, 30, dan 40 Mbps yang membuktikan bahwa bro IDS lebih baik dari segi *throughput* dengan snort IDS. Akan tetapi, pada percobaan dengan *background traffic* 10 Mbps dan 50 Mbps snort IDS lebih baik dari segi *throughput* dengan bro IDS. Salah satu pengaruh tinggi dan rendahnya *throughput* adalah proses dalam topologi yang dibuat. Penambahan snort IDS dan bro IDS menyebabkan pengurangan banyaknya paket data yang dikirimkan per satuan waktu. Snort IDS terdapat proses yang lebih kompleks yang bisa dilihat pada Gambar 1 dan pada proses deteksi snort IDS yang menggunakan *signature based detection* membuat proses deteksi yang lebih kompleks dikarenakan setiap *packet* yang melewati snort IDS harus mengalami proses pencocokan terhadap *string rules* yang terdapat di *database*. Sedangkan bro IDS terdapat proses yang lebih sederhana yang bisa dilihat pada Gambar 1 dan pada proses deteksi bro IDS yang menggunakan *anomaly based detection* membuat proses deteksi yang lebih sederhana dikarenakan setiap packet yang melewati bro IDS mengalami proses pencocokan *profile* packet terhadap *policy scripts*.

4.1.2 Delay



Gambar 6. Grafik Hasil Pengujian *Delay*

Dari hasil penelitian yang telah didapatkan pada Gambar 6 dapat dilihat hasil pengujian sistem terhadap parameter *delay*. Pada parameter *delay* semakin kecil *delay* dalam suatu jaringan maka semakin baik jaringan yang dibuat. Pengujian dilakukan dengan menerapkan snort IDS dan bro IDS dengan jumlah *rules* 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 yang disimulasikan pada jaringan dengan *background traffic* 0, 10, 20, 30, 40, 50 Mbps dari *bandwidth* yang ditetapkan = 100 Mbps.

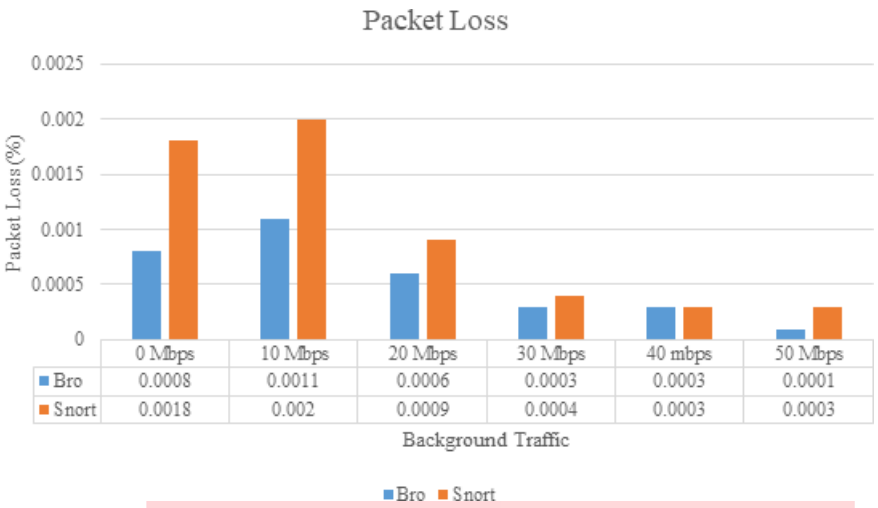
Rata-rata hasil parameter *delay* dengan *background traffic* 0 Mbps didapatkan hasil 0,00155 *seconds* untuk bro IDS dan 0,00168 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,00013 *seconds*. Rata-rata hasil parameter *delay* dengan *background traffic* 10 Mbps didapatkan hasil 0,00119 *seconds* untuk bro IDS dan 0,00120 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,00001 *seconds*.

Rata-rata hasil parameter *delay* dengan *background traffic* 20 Mbps didapatkan hasil 0,00109 *seconds* untuk bro IDS dan 0,00112 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,00003 *seconds*. Rata-rata hasil parameter *delay* dengan *background traffic* 30 Mbps didapatkan hasil 0,00106 *seconds* untuk bro IDS dan 0,00106 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS dan snort IDS memiliki hasil yang sama.

Rata-rata hasil parameter *delay* dengan *background traffic* 40 Mbps didapatkan hasil 0,00107 *seconds* untuk bro IDS dan 0,00107 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS dan snort IDS memiliki hasil yang sama. Rata-rata hasil parameter *delay* dengan *background traffic* 50 Mbps didapatkan hasil 0,00106 *seconds* untuk bro IDS dan 0,00105 *seconds* untuk snort IDS, dari hasil tersebut *delay* pada bro IDS lebih buruk daripada snort IDS dengan selisih 0,00101.

Dari enam percobaan yag dilakukan terdapat tiga percobaan dengan *background traffic* 0, 10, dan 20 Mbps yang membuktikan bahwa bro IDS lebih baik dari segi *delay* dengan snort IDS. Akan tetapi, pada percobaan dengan *background traffic* 50 Mbps membuktikan bahwa snort IDS lebih baik dari segi *delay* dengan bro IDS. Salah satu faktor tinggi atau rendahnya *delay* adalah proses dalam topologi yang dibuat. Penambahan snort IDS dan bro IDS menyebabkan pengaruh terhadap waktu pengiriman paket data dari awal pengiriman hingga tujuan. Snort IDS memiliki proses dan metode yang lebih kompleks pada infrastrukturnya daripada bro IDS yang dapat dilihat pada Gambar 1. Setiap proses yang terdapat pada infrastruktur IDS memiliki waktu untuk memprosesnya. Semakin banyak proses dalam infrastruktur yang terjadi akan membuat semakin lama waktu pengiriman *packet* ke tujuan. Snort IDS memiliki lima proses sedangkan bro IDS memiliki tiga proses dalam infrastrukturnya.

4.1.3 Packet Loss



Gambar 7. Grafik Hasil Pengujian *Packet Loss*

Dari hasil penelitian yang telah didapatkan pada Gambar 7 dapat dilihat hasil pengujian sistem terhadap parameter *packet loss*. Pada parameter *packet loss* semakin kecil *packet loss* dalam suatu jaringan maka semakin baik jaringan yang dibuat. Pengujian dilakukan dengan menerapkan snort IDS dan bro IDS dengan jumlah *rules* 10, 20, 30, 40, 50, 60, 70, 80, 90, dan 100 yang disimulasikan pada jaringan dengan *background traffic* 0, 10, 20, 30, 40, 50 Mbps dari *bandwidth* yang ditetapkan = 100 Mbps.

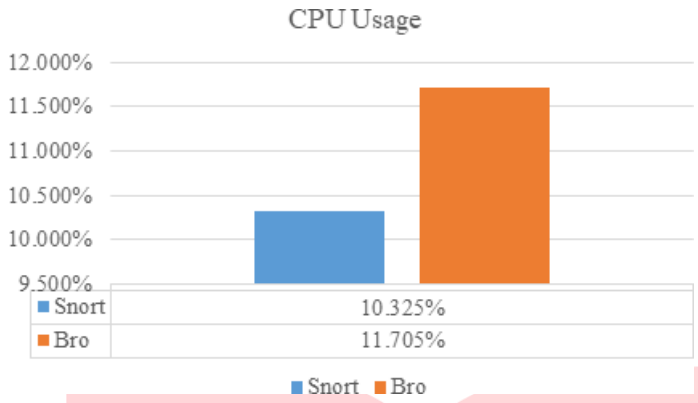
Rata-rata hasil parameter *packet loss* dengan *background traffic* 0 Mbps didapatkan hasil 0,0008% untuk bro IDS dan 0,0018% untuk snort IDS, dari hasil tersebut *packet loss* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0010%. Rata-rata hasil parameter *packet loss* dengan *background traffic* 10 Mbps didapatkan hasil 0,0020% untuk bro IDS dan 0,0011% untuk snort IDS, dari hasil tersebut *packet loss* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0009%.

Rata-rata hasil parameter *packet loss* dengan *background traffic* 20 Mbps didapatkan hasil 0,0006% untuk bro IDS dan 0,0009% untuk snort IDS, dari hasil tersebut *packet loss* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0003%. Rata-rata hasil parameter *packet loss* dengan *background traffic* 30 Mbps didapatkan hasil 0,0003% untuk bro IDS dan 0,0004% untuk snort IDS, dari hasil tersebut *packet loss* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0001%.

Rata-rata hasil parameter *packet loss* dengan *background traffic* 40 Mbps didapatkan hasil 0,0003% untuk bro IDS dan 0,0003% untuk snort IDS, dari hasil tersebut *packet loss* pada snort IDS dan bro IDS memiliki hasil yang sama. Rata-rata hasil parameter *packet loss* dengan *background traffic* 50 Mbps didapatkan hasil 0,0001% untuk bro IDS dan 0,0003% untuk snort IDS, dari hasil tersebut *packet loss* pada bro IDS lebih baik daripada snort IDS dengan selisih 0,0002%.

Dari enam percobaan yag dilakukan terdapat empat percobaan dengan *background traffic* 0, 10, 20, 30, dan 50 Mbps yang membuktikan bahwa bro IDS lebih baik dari segi *packet loss* daripada snort IDS. Salah satu faktor tinggi atau rendahnya *packet loss* adalah pada jenis traffic di jaringan. Pada penelitian ini menggunakan traffic TCP (*Transmission Control Protocol*) dengan *traffic generator*. TCP memiliki sifat yang *reliable* dalam pengiriman paket datanya, jika tidak ada packet *acknowledgement* dalam suatu rentan waktu dari penerima maka segmen TCP akan mengirimkan transmisi ulang. Hubungan dari pengiriman *packet* TCP dengan proses yang terjadi pada topologi yang dibuat sangat erat. Dengan penerapan snort IDS yang memiliki infrastruktur dan metode yang lebih kompleks dibandingkan bro IDS menyebabkan waktu pengiriman paket data yang lebih lama. Semakin lama proses pengiriman paket TCP ke tujuan akan menyebabkan TCP akan mengirimkan transmisi ulang dan dapat dikatakan *packet loss* terjadi.

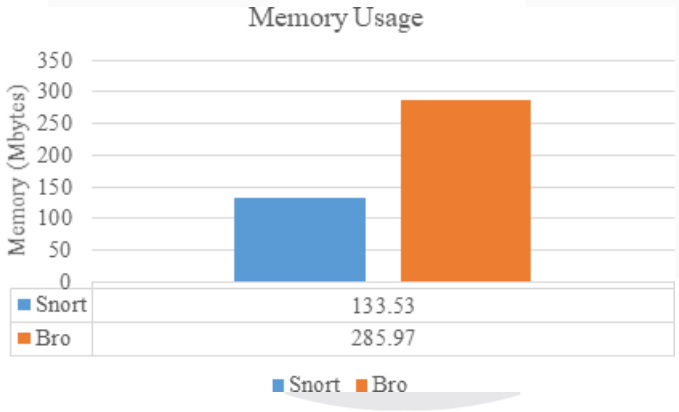
4.1.4 CPU *Usage*



Gambar 8. Grafik Hasil Pengujian CPU *Usage*

Dari hasil penelitian yang telah didapatkan pada Gambar 8 dapat dilihat hasil penggunaan CPU dalam menerapkan snort IDS dan bro IDS. Pengujian dilakukan pada *Personal Computer* (PC) yang memiliki spesifikasi CPU atau processor Intel(R) Core (TM) i3-3217U CPU @ 1.80 GHz. Pada parameter CPU *usage* semakin besar CPU *usage* maka program atau *tools* yang dijalankan pada sebuah PC membutuhkan sebuah *resource* CPU yang lebih besar. Pada simulasi yang dijalankan snort IDS dan bro IDS tidak menambahkan *rules* atau *scripts* untuk proses deteksi. Rata-rata hasil parameter CPU *usage* didapatkan 10,325% untuk snort IDS dan 11,705% untuk bro IDS, dari hasil tersebut CPU *usage* pada bro IDS lebih besar daripada snort IDS. Faktor yang mempengaruhi besar maupun kecilnya CPU *usage* adalah seberapa besar atau banyaknya program atau *tools* yang dijalankan. Pengaruh CPU *usage* pada jaringan sangat berdampak jika CPU *usage* pada sebuah *host* dalam suatu jaringan memiliki persentase yang mendekati atau melebihi batas maksimal (100%). Jikalau CPU *usage* pada suatu *host* dalam suatu jaringan memiliki CPU usage dalam batas normal maka jaringan tersebut tidak mendapatkan dampaknya.

4.1.5 Memory Usage



Gambar 9. Grafik Hasil Pengujian Memory *Usage*

Dari hasil penelitian yang telah didapatkan pada Gambar 9 dapat dilihat hasil penggunaan *memory* dalam menerapkan snort IDS dan bro IDS. Pengujian dilakukan pada *Personal Computer* (PC) yang memiliki spesifikasi *memory* 5,89 GB. Pada parameter *memory usage* semakin besar *memory usage* maka program atau *tools* yang dijalankan pada sebuah PC membutuhkan *resource memory* yang lebih besar. Pada simulasi yang dijalankan snort IDS dan bro IDS tidak menambahkan *rules* atau *scripts* untuk proses deteksi. Rata-rata hasil parameter *memory usage* didapatkan 133,53 MB untuk snort IDS dan 285,97 MB untuk bro IDS, dari hasil tersebut *memory usage* pada bro IDS lebih besar daripada snort IDS. Faktor yang mempengaruhi besar maupun kecilnya *memory usage* adalah seberapa besar atau banyaknya program atau *tools* yang dijalankan. Pengaruh *memory usage* pada jaringan sangat berdampak jika *memory usage* pada sebuah *host* dalam suatu jaringan melebihi batas maksimal. Jikalau

memory usage pada suatu *host* dalam suatu jaringan memiliki *memory usage* dalam batas normal maka jaringan tersebut tidak mendapatkan dampaknya.

5. Kesimpulan

Berdasarkan pengujian yang telah dilakukan dapat disimpulkan bro IDS lebih baik dalam parameter *throughput* dengan perbandingan hasil pengujian 4:1 dengan snort IDS. Bro IDS lebih baik pada saat pengujian dengan menggunakan *background traffic* 0, 20, 30, dan 40 Mbps. Pada parameter *delay* dapat disimpulkan bro IDS lebih baik daripada snort IDS dengan perbandingan hasil pengujian 3:1. Bro IDS lebih baik pada saat pengujian dengan menggunakan *background traffic* 0, 10, dan 20 Mbps. Pada parameter *packet loss* dapat disimpulkan bro IDS lebih baik daripada snort IDS dengan perbandingan hasil pengujian 5:1. Bro IDS lebih baik pada saat pengujian dengan menggunakan *background traffic* 0, 10, 20, 30, dan 50 Mbps. Alasan dari bro IDS lebih baik daripada snort IDS dikarenakan terdapat faktor metode deteksi yang diterapkan dan infrasturktur pada masing-masing *tools* IDS. Sedangkan dalam parameter CPU *usage* dan *memory usage* bro IDS membutuhkan *resource* yang lebih besar dibandingkan snort IDS. Bro IDS membutuhkan *resource* CPU sebesar 11,705% dan *resource memory* sebesar 285,97 MB. Sedangkan snort IDS hanya membutuhkan *resource* CPU sebesar 10,325% dan *resource memory* sebesar 133,53 MB. Hasil penelitian menghasilkan data yang hampir mendekati hasil sebenarnya walaupun terdapat beberapa data yang memiliki hasil yang tidak sesuai dengan hasil yang seharusnya dikarenakan penelitian dilakukan dalam simulasi mininet dan penggunaan *traffic generator*. Diharapkan untuk menggunakan *real device* dan traffic yang sebenarnya untuk penyempurnaan penelitian selanjutnya. Pengujian dengan menggunakan *high speed network* juga sangat dianjurkan dalam membandingkan snort IDS dan bro IDS untuk penelitian selanjutnya.

Daftar Pustaka

[1] A. Alhomoud, R. Munir, J. P. Disso, I. Awan, and A. Al-Dhelaan. Performance evaluation study of intrusion detection systems. *Procedia Computer Science*, 5(Supplement C):173 – 180, 2011. The 2nd International Conference on Ambient Systems, Networks and Technologies (ANT-2011) / The 8th International Conference on Mobile Web Information Systems (MobiWIS 2011).

[2] D. A. Bhosale and V. M. Mane. Comparative study and analysis of network intrusion detection tools. In *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, pages 312–315, Oct 2015.

[3] Bro. Bro NIDS Tools, year = 2014, url = https://bro.org/, access date = 2018-8-5.

[4] Calyptix. Top 7 Network Attack Types in 2015 Source McAfee Labs, Calyptix Security, year = 2015, url = https://www.calyptix.com/top-threats/top-7-network-attack-types-in-2015-so-far/, access date = 2018-8-2.

[5] Calyptix. Top 7 Network Attack Types in 2016 Source McAfee Labs, Calyptix Security, year = 2016, url = https://www.calyptix.com/top-threats/top-7-network-attack-types-2016/, access date = 2018-8-2.

[6] Calyptix. Top 7 Network Attack Types in 2017 Source McAfee Labs, Calyptix Security, year = 2017, url = https://www.calyptix.com/top-threats/top-8-network-attacks-type-2017/, access date = 2018-14-8.

[7] Cisco. Quality of Service Networking, year = 2012, url = http://docwiki.cisco.com/wiki/Quality_of_Service Networking, access date = 2018-8-6.

[8] S. Cooper. 10 Top Intrusion Detection Tools for 2018, year = 2018, url = https://www.comparitech.com/net-admin/network-intrusion-detection-tools/gref, access date = 2018-8-9.

[9] A. Deepa and V. Kavitha. A comprehensive survey on approaches to intrusion detection system. *Procedia Engineering*, 38(Complete):2063–2069, 2012.

[10] T. A. Fauzi. *Integrasi Intrusion Detection System pada Software Define Network*. Telkom University, 2016.

[11] O. N. Foundation. *SDN Architecture Overview*. Open Networking Foundation, 2013.

[12] J. Ibrahim and S. Gajin. Sdn-based intrusion detection system. *Infoteh-Jahorina*, 16:621–624, 2017.

[13] G. Khalil. Open source ids high performance shootout. *SANS Institute InfoSec Reading Room*, (6):113–125, February 2015.

[14] R. Pratama. *Perancangan dan Implementasi Adaptive Intrusion Prevention System (IPS) untuk Pencegahan Penyerangan pada Arsitektur Software Define Network (SDN)*. Telkom University, 2017.

[15] L. Ran, F. Taiyi, G. Yunfeng, L. Yang, Cong, H. Yang, and D. Huilong. The research of openflow management and control interface protocols based on sdn technology. In *2015 IEEE International Conference on Computer and Communications (ICCC)*, pages 45–49, Oct 2015.

[16] Snort. Snort NIDS Tools, year = 2017, url = https://snort.org/, access date = 2018-8-2.

[17] S.Vijayarani and M. S. S. Intrusion detection system-a study. *International Journal of Security, Privacy and Trust Management (IJSPTM)*, 4(1):31–44, feb 2015.

[18] T. Wang and H. Chen. Sguard: A lightweight sdn safe-guard architecture for dos attacks. *China Communications*, 14(6):113–125, 2017.

[19] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie. A survey on software-defined networking. *IEEE Communications Surveys Tutorials*, 17(1):27–51, Firstquarter 2015.